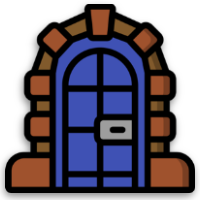
ADVENTURE IN THE JAVA

Player

Monsters



Dungeons

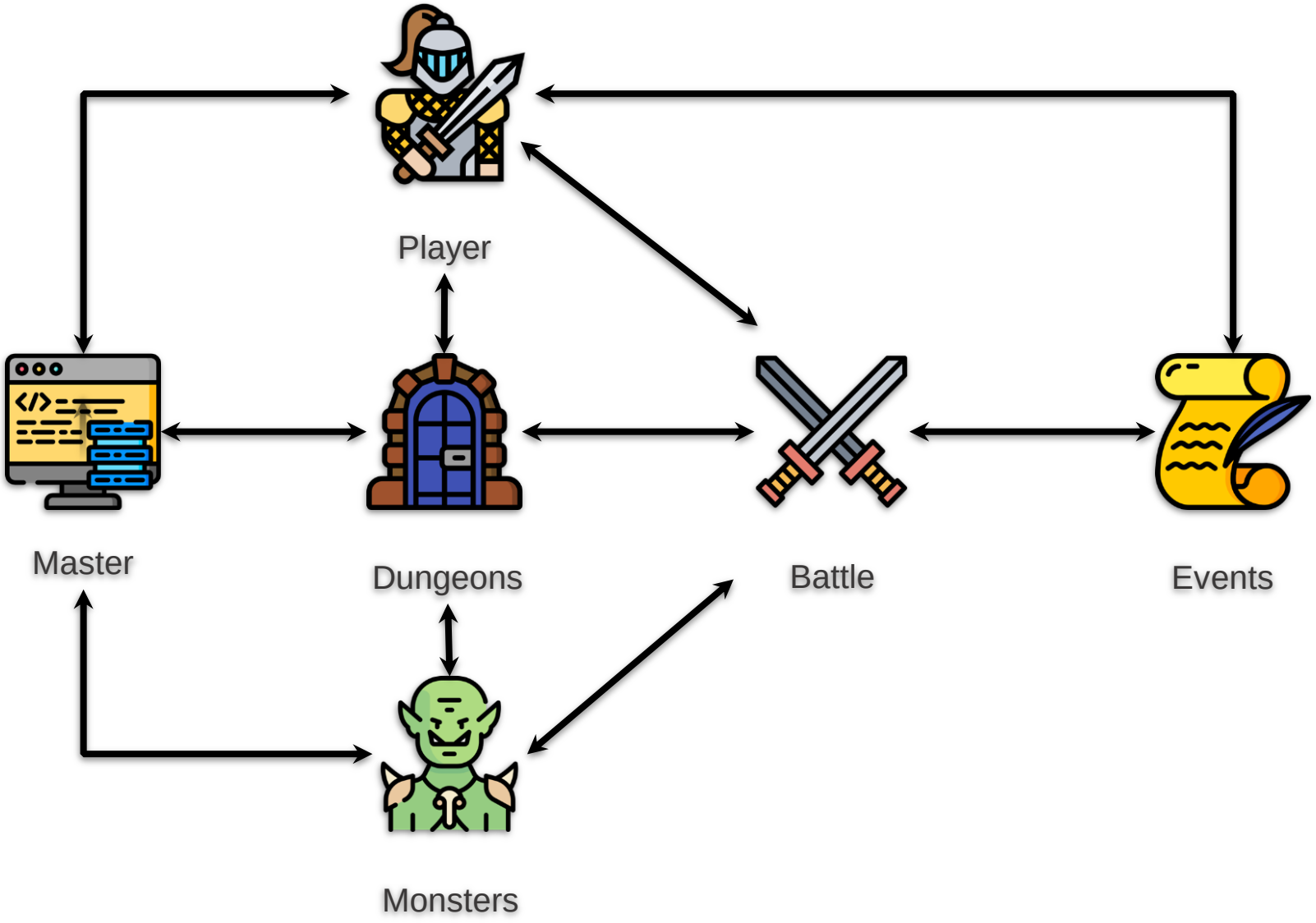
Battle

Events

Team J. J. J
팀원 : 김승연 / 원윤경 / 양창일 / 양미선 / 지용욱

J. J. J ’s TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말



Ln 1, Col 1 100% Windows (CRLF) UTF-8

J. J. J ’s TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

PLAYER

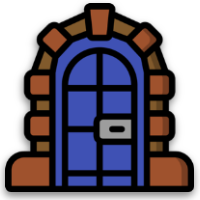
01

Monsters

02

Battle

03



Dungeons

04

Events

05

Ln 1, Col 1 100% Windows (CRLF) UTF-8

01

Player

양창일

PLAYER

- 게임 시작 시 사용자의 캐릭터를 만들어주는 기능

- 초기 status값 interface로 상수처리 예정

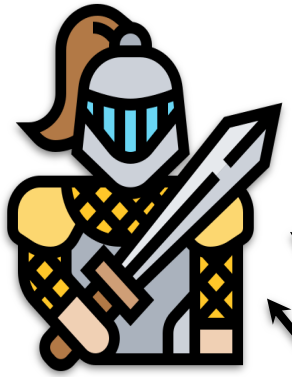
- player 성장 시, 설정 수식에 따른
  status 수치 변경/저장기능

PLAYER ver2
extends PLAYER

++장비, 포션, 스킬, 상점;

++ArrayList사용으로 구매 정보 관리;

++ 포션 및 스킬의 사용 메서드;



PLAYER

EQUIPMENT

구매

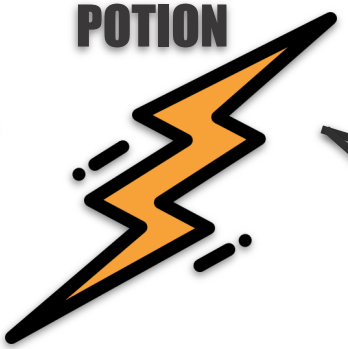
판매

장착

POTION

구매

사용

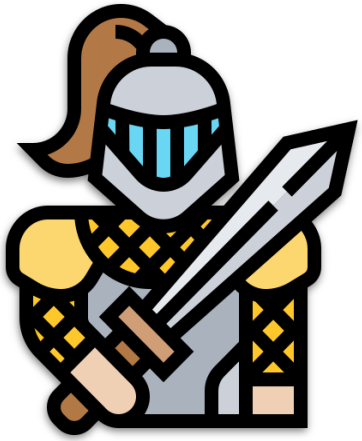


SKILL

구매

사용

ENTITY



PLAYER

```java
// 경험치 값을 확인해 레벨업 및 스테이터스 업데이트를 하는 메서드
void checkLevelUp() {
    calEquipStat();
    while (true) {
        if (getCurrentExp() >= levelUpExp) {

            currentExp = currentExp - levelUpExp;

            currentLevel += 1;

            levelUpExp = (int) (levelUpExp * 1.3f);
            setMaxHealth((int) (getMaxHealth() * 1.3f) / 1);
            setCurrentHealth(getMaxHealth());
            setCurrentStrength((int) (getCurrentStrength() * 1.3f));
            setEvasion(getEvasion() + 1);
            invenCurrentStrength = currentStrength + inven.equipPower;
            invenMaxHealth = maxHealth + inven.equipHealth;
            invenCurrentHealth = currentHealth + inven.equipHealth;
            invenCurrentEvasion = getEvasion() + inven.equipEvasion;
            System.out.println("     ┌──────Congraturations!!!!──────┐");
            System.out.println("     |    레벨업 하였습니다!         |");
            System.out.println("     └──────────────────────────────────┘");
            System.out.println("         > 레벨 : " + this.currentLevel + " UP↑    <");
            System.out.println("         > HP : " + invenCurrentHealth + "/" + invenMaxHealth + "<");
            System.out.println("         > 공격력 : " + invenCurrentStrength + "    <");
            System.out.println("         > 회피율 : " + invenCurrentEvasion + "%    <");
            System.out.println("         > EXP : " + this.currentExp + "/" + this.levelUpExp + " <");
            System.out.println("     └──────────────────────────────────┘");

            if (getCurrentExp() < levelUpExp) {
                break;
            }
        } else {
            break;
        }
```

```java
// 캐릭터의 이름을 받는 메서드
void addName() {

    super.setName(JOptionPane.showInputDialog(" 캐릭터의 이름을 입력해주세요."));

    while (true) {
        System.out.println("    입력하신 이름이 " + super.getName() + "님이 맞습니까? 맞으면 y 틀리면n");
        String check = sc.nextLine();

        if (check.equals("y") || check.equals("Y")) {
            System.out.println("    캐릭터가 생성되었습니다!");
            break;

        } else if (check.equals("n") || check.equals("N")) {
            addName();
            break;
        } else {
            System.out.println("    잘못 누르셨습니다.");
            continue;
        }
    }
}
```

J. J. J ’s TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

items
- A_Hat.java
- A_HeadPiece.java
- B_DiamondArmor.java
- B_OldArmor.java
- B_ShiningArmor.java
- C_InvisibilityCloak.java
- C_OldCloak.java
- C_ShiningCloak.java
- D_DiamondWand.java
- D_GoldWand.java
- D_SilverWand.java
- Inven.java
- Item.java

```java
// 장비의 스탯 계산
public void calEquipStat() {
    inven.equipHealth = 0;
    inven.equipPower = 0;
    inven.equipEvasion = 0;
    for (int i = 0; i < inven.equip.size(); i++) {

        inven.equipPower += inven.equip.get(i).attackPower;
        inven.equipHealth += inven.equip.get(i).health;
        inven.equipEvasion += inven.equip.get(i).evasion;

    }

}
```

EQUIPMENT

```java
public void equipItem() {

    System.out.println("    ======================================");
    System.out.println("    장착할 장비를 골라주세요.");
    System.out.println("    ======================================");

    System.out.println("    0. 마을로 돌아가기");

    int select = sc.nextInt();

    sc.nextLine();

    if (select == 0) {
        return;
    }

    inven.checkType(inven.inven.get((select - 1)).equipmentType); // 장비 타입 비교해서 중복된 타입일 시 장비 반환

    inven.equip.add(inven.inven.get((select - 1)));

    System.out.println(inven.inven.get((select - 1)).equipmentName + "장착");

    inven.inven.remove((select - 1));

    int dmg = invenMaxHealth - invenCurrentHealth;

    calEquipStat();
    System.out.println("    + 장비 공격력 : " + inven.equipPower + ", " + "+ 장비 체력 : " + inven.equipHealth + ", "
            + "+ 장비 회피율 : " + inven.equipEvasion);
    invenCurrentStrength = getCurrentStrength() + inven.equipPower;
    invenMaxHealth = getMaxHealth() + inven.equipHealth;
    invenCurrentHealth = getCurrentHealth() + inven.equipHealth - dmg;
    invenCurrentEvasion = getEvasion() + inven.equipEvasion;

    inven.showInventory();
    inven.showEquip();

}
```

```java
// 장비타입을 비교해서 인벤토리로 장착하던 장비 반환
public void checkType(int checkNum) {

    for (int i = 0; i < equip.size(); i++) {

        if (checkNum == equip.get(i).equipmentType) {

            // check = checkNum;
            inven.add(equip.get(i));
            equip.remove(i);
            break;
        }

    }

}
```

```java
public void buyEquipment(int select) {
    switch (select) {
    case 1:

        if (A1.gold > gold) {
            System.out.println("골드가 부족하여 구매할 수 없습니다.");
            break;
        }
        gold = gold - A1.gold;
        inven.addEquipment(A1);
        System.out.println("장비를 구매하였습니다.");

        break;
```

```java
public void sellEquipment() {

    System.out.println("======================================");
    System.out.println("판매할 장비를 골라주세요.");
    System.out.println("======================================");

    System.out.println("0. 마을로 돌아가기");

    int select = sc.nextInt();

    sc.nextLine();

    if (select == 0) {
        return;
    }
    setGold(getGold() + inven.inven.get(select - 1).gold);
    inven.inven.remove(select - 1);

    System.out.println("판매 되었습니다.");

}
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

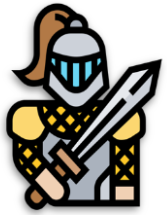
```java
// 포션구매 메서드
public void buyPotion(int i, int num) {

    // 처음에만 포션들을 추가
    if (potion.size() == 0) {
        potion.add(sp);
        potion.add(np);
        potion.add(bp);
    }

    switch (i) {
    case 1:
        if (potion.get(0).price * num > gold) {
            System.out.println("골드가 부족하여 구매할수없습니다.");
            break;
        }
        potion.set(0, new Potion("소형 체력 물약", 30, (potion.get(0).pNum) + num, 20));

        gold = gold - 20 * num;
        System.out.println(potion.get(i - 1).pName + ", " + potion.get(i - 1).pNum);

        System.out.println(potion.toString());
        System.out.println("포션을 구매하였습니다.");

        break;
```

POTION

```java
// 포션 사용 메서드
public void usePotion(int i) {

    switch (i) {
    case 1:
        potion.set(0, new Potion("소형 체력 물약", 30, (potion.get(0).pNum) - 1, 20));

        invenCurrentHealth = invenCurrentHealth + 30;

        if (invenCurrentHealth > invenMaxHealth) {
            invenCurrentHealth = invenMaxHealth;
        }

        break;
```

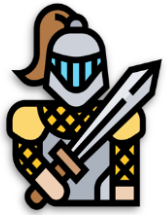
SKILL

```
public class SkillInven {

    public ArrayList<Skill> skill = new ArrayList<Skill>(3);
    Scanner sc = new Scanner(System.in);

    public void buySkill(Player p, Skill s) {

        if (s.gold > p.getGold()) {
            System.out.println("골드가 부족하여 구매할수없습니다.");
            return;
        }

        for (int i = 0; i < skill.size(); i++) {

            if (s.skillName.equals(skill.get(i).skillName)) {
                System.out.println("이미 가지고 있는 스킬입니다.");
                return;
            }
        }
        skill.add(s);

        p.setGold(p.getGold() - s.gold);

    }

    public Skill useSkill(Player p, int select) {

        skill.set(select, new Skill(skill.get(select).skillName, skill.get(select).numOfChance - 1,
                skill.get(select).multiple, skill.get(select).gold));

        Skill s = skill.get(select);

        return s;

    }

    public void resetSkillChance() {

        for (int i = 0; i < skill.size(); i++) {
            skill.set(i, new Skill(skill.get(i).skillName, 3, skill.get(i).multiple, skill.get(i).gold));
        }
    }



    public void showSkill() {
        for (int i = 0; i < skill.size(); i++) {

            System.out.println((i + 1) + ". " + skill.get(i).toString());

        }

    }
}
```



PLAYER

```
public class 어려웠던점 {

    void 장비능력치() {
        System.out.println(
        "    하지만 체력정보를 관리할 때 체력과 장비능력치를 합쳐서 초기화를 하게되면 \n\t" +
        "   체력이 오르는 문제가 있어서 초기화를 하는 시기가 중요한데 \n\t" +
        "   장비를 장착한 후 와 레벨업을 한 후에만 해주어야 했다 \n\t"

                );

    }

    void 메서드위치() {
        System.out.println(
        "    메서드의 위치및 분류가 어려웠다" );
    }

}
```



PLAYER

```
public class 개선할점 {


    void 1.변수선언() {
        System.out.println(
         "많은 변수 선언 -> 소통은 x 정리x(혼잡)");
    }
    void 2.틀() {
        System.out.println(
        "추가하기전에 틀을 더 잘 잡았어야한다");
    }
    void 3.테스트() {
        System.out.println(
        "정보가 많아서 오류테스트를 할게 많은데 테스트가 부족한채로 커밋해서 혼란");
    }
    void 4.파악() {
        System.out.println(
        "욕심만 많아서 해놓은 것을 검토하지 않고 기능만 늘리려 했다");
    }
    void 5.주석() {
        System.out.println(
        "주석달기");
    }
}
```

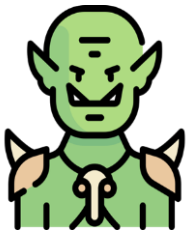
02

Monster

원윤경

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말



Monsters

엔티티

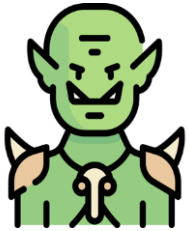
Entity.java

몬스터

Monster.java

- 쥐 — Rat.java
- 닭 — Chicken.java
- 토끼 — Rabbit.java
- 개 — Dog.java
- 원숭이 — Monkey.java
- 양 — Sheep.java
- 돼지 — Pig.java
- 뱀 — Snake.java
- 말 — Horse.java
- 소 — Cow.java
- 호랑이 — Tiger.java
- 용 — Dragon.java

Ln 1, Col 1 100% Windows (CRLF) UTF-8



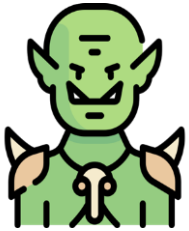
Monsters

엔티티

Entity.java

```
public void initCurrentStats() {
    currentHealth = baseHealth;
    currentStrength = baseStrength;
    currentDefense = baseDefense;
}
```

```
public class Entity {
    String name;
    int evasion;
    int goldWorth;
    int expWorth;

    int baseHealth, currentHealth, maxHealth;
    int baseStrength, currentStrength;
    int baseDefense, currentDefense;
    int currentExp;
    int currentLevel;
    int gold;
```



몬스터

Monsters

Monster.java

```java
public class Monster extends Entity {
    private int defense;
    private int weakness;
    private int stage;
    Random rand;

    protected Monster() {
        defense = EnemyBasics.BASE_DEFENSE;
        weakness = 0;
        stage = EnemyBasics.STAGE;
        rand = new Random();
    }
```

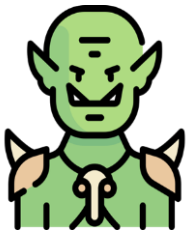
상수 처리

EnemyBasics.java

```java
public interface Rounds {
    int first = 1, second = 2, third = 3, forth = 4,
        fifth = 5, sixth = 6, seventh = 7, eighth = 8, nineth = 9,
        tenth = 10, eleventh = 11, twelveth = 12;
}
```

```java
public Monster makeMonster(int stage) {
    Monster monster = null;

    switch (stage) {
    case Rounds.first:
        monster = new Rat();
        break;
    case Rounds.second:
        monster = new Chicken();
        break;
    case Rounds.third:
        monster = new Rabbit();
        break;
    case Rounds.forth:
        monster = new Dog();
        break;
    case Rounds.fifth:
        monster = new Monkey();
        break;
    case Rounds.sixth:
        monster = new Sheep();
        break;
    case Rounds.seventh:
        monster = new Pig();
        break;
    case Rounds.eighth:
        monster = new Snake();
        break;
    case Rounds.nineth:
        monster = new Horse();
        break;
    case Rounds.tenth:
        monster = new Cow();
        break;
    case Rounds.eleventh:
        monster = new Tiger();
        break;
    case Rounds.twelveth:
        monster = new Dragon();
        break;
    }
    return monster;
}
```

다형성



몬스터

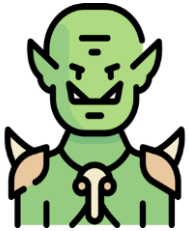
Monsters

Monster.java

```
public void setBaseHealth(int stage) {
    super.setBaseHealth(EnemyBasics.BASE_HEALTH + stage * 10);
}

public void setBaseStrength() {
    baseStrength = EnemyBasics.BASE_STRENGTH + stage * 10;
}
```

```
public void setEvasion() {
    evasion = (rand.nextInt(100) + 1);
}

public int getGoldWorth() {
    return goldWorth;
}

public void setGoldWorth(int exp) {
    goldWorth = exp;
    int gold_max = exp * 5;
    int gold_min = exp * 1;
    goldWorth = rand.nextInt(gold_max - gold_min + 1) + gold_min;
}

public int getExpWorth() {
    return expWorth;
}

public void setExpWorth(int stage) {
    int exp_max = (int) (stage * 10);
    int exp_min = (int) (stage * 5);
    expWorth = stage * 10 + rand.nextInt(exp_max - exp_min + 1) + exp_min;
}
```

```
public void setName(String name) {
    String[] kinds = { "기본 ", "날쌘 ", "덩치가 큰 ", "이빨이 날카로운 ", "알 수 없는 " };
    Random rand = new Random();
    int randIndex = rand.nextInt(5);

    this.name = kinds[randIndex] + name;
    title = kinds[randIndex];

}
```



몬스터들

Monsters

EnemyBasics.java

```
public interface EnemyBasics {
    int STAGE = 1, BASE_HEALTH = 100, BASE_STRENGTH = 20,
    BESE_EVASION = 20, BASE_GOLD_WORTH = 100,
    BASE_EXP_WORTH = 20, BASE_DEFENSE = 10;
}
```

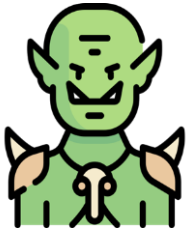
Entity.java

```
public class Chicken extends Monster {

    public Chicken() {
        setStage(2);
        setName("닭");
        setBaseHealth(getStage() + 2);
        setBaseStrength();
        setExpWorth(getStage());
        setGoldWorth(getExpWorth());
        setWeakness(4);
        setEvasion(0);//
        initCurrentStats();
    }
```

```
public class Dragon extends Monster {
    public Dragon() {
        setStage(12);
        setName("용");
        setBaseHealth(getStage() + 380);
        setBaseStrength();
        setExpWorth(getStage());
        setGoldWorth(getExpWorth());
        setWeakness(30);
        setEvasion(5);
        initCurrentStats();
    }
}
```

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말



Monsters

몬스터

Dungeon.java

```
Monster makeMonsters(int stage) {
    Random rand = new Random();
    int numOfMonsters = rand.nextInt(5) + 1;
    ArrayList<Monster> monsters = new ArrayList<>(numOfMonsters);

    Monster randMonster = new Monster();

    int randIndex = rand.nextInt(numOfMonsters);
    int randValue = rand.nextInt(20) + 10;
    for (int i = 0; i < numOfMonsters; i++) {
        m = m.makeMonster(stage);
        if (m.title.equals("놀렘 ")) {
            m.setEvasion(m.getEvasion() + randValue);
        } else if (m.title.equals("덩치가 큰 ")) {
            m.setCurrentHealth(m.getCurrentHealth() + randValue);
        } else if (m.title.equals("이빨이 날카로운 ")) {
            m.setCurrentStrength(m.getCurrentStrength() + randValue);
        } else if (m.title.equals("알 수 없는 ")) {
            m.setEvasion(m.getEvasion() + randValue);
            m.setCurrentHealth(m.getCurrentHealth() + randValue);
            m.setCurrentStrength(m.getCurrentStrength() + randValue);
        }
        monsters.add(m);
    }
    randMonster = monsters.get(randIndex);
    System.out.println("\n" + "  " + m.getName() + "을/를 만났습니다");
    m.showData();

    return randMonster;
}
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

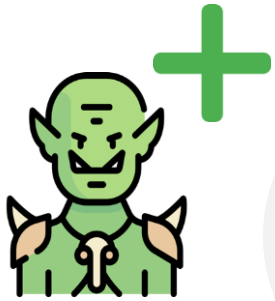
플레이어 정보 저장

추가기능

```
public void savePlayer() {
    FileOutputStream f = null;
    ObjectOutputStream oos = null;
    try {
        this.showStatus();
        f = new FileOutputStream("data.ser");
        oos = new ObjectOutputStream(f);
        oos.writeObject(this.getName());
        oos.writeObject(this.getCurrentLevel());
        oos.writeObject(this.invenCurrentHealth);
        oos.writeObject(this.getMaxHealth());
        oos.writeObject(this.getCurrentStrength());
        oos.writeObject(this.getEvasion());
        oos.writeObject(this.getCurrentExp());
        oos.writeObject(this.getLevelUpExp());
        oos.writeObject(this.getGold());
        oos.writeObject(this.invenCurrentHealth);
        oos.writeObject(this.invenMaxHealth);
        oos.writeObject(this.invenCurrentStrength);
        oos.writeObject(this.invenCurrentEvasion);
        oos.writeObject(this.bossCount);
        oos.writeObject(this.stage2Count);
        oos.writeObject(this.stage3Count);
```

Player.java

```
        f.close();
        oos.close();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (f != null)
            try {
                f.close();
            } catch (IOException e) {
            }
        if (oos != null)
            try {
                oos.close();
            } catch (IOException e) {
            }

    }
    System.out.println("플레이어 정보가 저장되었습니다");
}
```



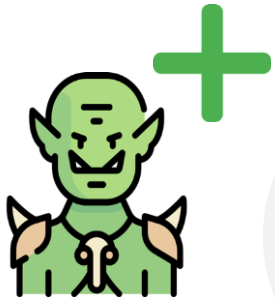
플레이어 정보 불러오기

추가기능

```
public void loadPlayer() {
    FileInputStream f = null;
    ObjectInputStream ois = null;
    String name;
    int currentLevel, currentHealth, maxHealth, currentStrength,
    evasion, exp, gold,levelUpExp;
    int invenCurrentHealth,invenMaxHealth,invenCurrentStrength,
    invenCurrentEvasion;
    int bossCount,stage2Count,stage3Count;
    try {
        f = new FileInputStream("data.ser");
        ois = new ObjectInputStream(f);
        name = ((String) ois.readObject());
        currentLevel = ((Integer) ois.readObject());
        currentHealth = ((Integer) ois.readObject());
        maxHealth = ((Integer) ois.readObject());
        currentStrength = ((Integer) ois.readObject());
        evasion = ((Integer) ois.readObject());
        exp = ((Integer) ois.readObject());
        levelUpExp = ((Integer) ois.readObject());//추가
        gold = ((Integer) ois.readObject());
        invenCurrentHealth = ((Integer) ois.readObject());
        invenMaxHealth = ((Integer) ois.readObject());
        invenCurrentStrength = ((Integer) ois.readObject());
        invenCurrentEvasion = ((Integer) ois.readObject());
        bossCount = ((Integer) ois.readObject());
        stage2Count = ((Integer) ois.readObject());
        stage3Count = ((Integer) ois.readObject());
```

Player.java

```
        this.setName(name);
        this.setCurrentLevel(currentLevel);
        this.setCurrentHealth(currentHealth);
        this.setMaxHealth(maxHealth);
        this.setCurrentStrength(currentStrength);
        this.setEvasion(evasion);
        this.setCurrentExp(exp);
        this.setLevelUpExp(levelUpExp);
        this.setGold(gold);
        this.invenCurrentHealth =invenCurrentHealth;
        this.invenMaxHealth = invenMaxHealth;
        this.invenCurrentStrength =invenCurrentStrength;
        this.invenCurrentEvasion =invenCurrentEvasion;
        this.bossCount=bossCount;
        this.stage2Count=stage2Count;
        this.stage3Count=stage3Count;

        ois.close();
        System.out.println("플레이어 정보를 불러옵니다");
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (f != null)
            try {
                f.close();
            } catch (IOException e) {
            }
        if (ois != null)
            try {
                ois.close();
            } catch (IOException e) {
            }
    }
}
```



Monster.java

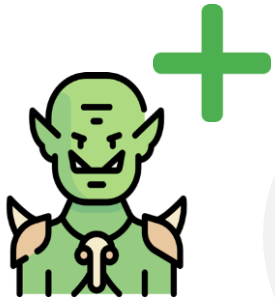
문제 및 해결 방법

문제

- 코드 통합시 변수명/메서드명 불일치
- 변수와 getter/setter 메서드의 혼용
- 객체 직렬화/역직렬화 시 객체 타입
- 잘못된 변수 출력 오류

해결 방법

- 변수명/메서드명 변경
- 변수/메서드 형식 통일화
- 변수에 맞는 타입으로 직렬화
- 출력 전에 변수명 확인



Monster.java

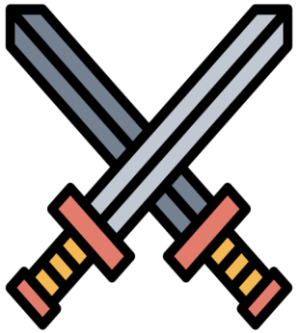
개선점

- 인터페이스/추상클래스 구현
- 멀티 스레드
- 데이터베이스 정보 저장 및 관리
- 협업시 팀워크 및 소통

03

Battle

양미선



Battle

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

공격 회피

Battle

```
void playerAttack(Player p, Monster m) {

    damage = (p.invenCurrentStrength) * 3;

    if (monsterEvasion(m)) {
        System.out.println("    " + m.getName() + " 이(가) 플레이어의 공격을 회피했습니다! 데미지가 0이 됩니다.");
        damage = 0;
    }
    m.setCurrentHealth(m.getCurrentHealth() - damage);
    System.out.println("    ┌──────────────────────────────────────────────────┐");
    System.out.println("    >" + p.getName() + " 님이\n" + " >" + m.getName() + " 에게 " + damage
            +
    System.ou
}
```

```
boolean monsterEvasion(Monster m) {

    if ((rand.nextInt(100)) <= m.getEvasion()) {
        return pass = true;
    } else {
        return pass = false;
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

방어

Battle

```
void playerDefense(Player p, Monster m) {

    monsterDamage = m.getCurrentStrength();
    playerDefense = (int) ((Math.random() * m.getCurrentStrength())
                                          + p.getCurrentLevel());
    depenseDmg = monsterDamage - playerDefense;
    p.invenCurrentHealth = p.invenCurrentHealth - depenseDmg;
```

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

실행

Battle

```
int choicePlayerMovement(Monster m, Player p) {
    int result = 0;
    this.battleResult = result;

    while (true) {
        System.out.println("                                    ");
        System.out.println("      행동을 선택해 주세요.         ");
        System.out.println("                                    ");
        System.out.println();
        System.out.println
("   >1.공격       >3.방어   \n   >2.스킬 사용    >4.포션 사용    \n   >5. 도망가기(마을로)");
        int choice = Integer.parseInt(bt.nextLine());
        switch (choice) {
        case 1:
            playerAttack(p, m);
            monsterAttack(p, m);
            if (p.invenCurrentHealth <= 0) {
                result = 1;
                break;
            } else if (m.getCurrentHealth() <= 0) {
                break;
            } else {
                continue;
            }
```

```
case 5:
    result = 2;
    break;
}

return result;
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

실행

Battle

```
int choicePlayerMovement(Monster m, Player p) {
    int result = 0;
    this.battleResult = result;

    while (true) {
        System.out.println("                                    ");
        System.out.println("      행동을 선택해 주세요.         ");
        System.out.println("                                    ");
        System.out.println();
        System.out.println
("   >1.공격       >3.방어   \n   >2.스킬 사용    >4.포션 사용    \n   >5. 도망가기(마을로)");
        int choice = Integer.parseInt(bt.nextLine());
        switch (choice) {
        case 1:
            playerAttack(p, m);
            monsterAttack(p, m);
            if (p.invenCurrentHealth <= 0) {
                result = 1;
                break;
            } else if (m.getCurrentHealth() <= 0) {
                break;
            } else {
                continue;
            }
```

```
case 5:
    result = 2;
    break;
}

return result;
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

쓰레드

Battle

```
public boolean takeDie(Player p) {

    if (p.invenCurrentHealth <= 0) {
        // 새로운 쓰레드
        Thread death = new Thread() {
            @Override
            public void run() {
                for (int i = 20; i > 0; i--) {
                    System.out.println(i);
                    try {
                        sleep(1000);
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            }
        };
```

익명클래스

```
death.start();
int n;
while (true) {
    String choice = JOptionPane.showInputDialog(
            "   플레이어가 사망했습니다. \n\n ((주의!)) 20초 안에 입력해주세요."
          + "(입력 없으면 자동 종료) \n 1. 긴급마을이송 : 비용 - lv * 100 gold \n 2.게임 종료");
    try {
        n = Integer.parseInt(choice);
        if (!(n > 0 && n < 3)) {
            System.out.println("    정상적인 메뉴 선택이 아닙니다.\n메뉴를 다시 선택해주세요.");
            continue;
        }

    } catch (NumberFormatException e) {
        System.out.println("    잘못입력하셨습니다. 다시 선택해 주세요.");
        continue;
    } catch (InputMismatchException e) {
        System.out.println("    잘못입력하셨습니다. 다시 선택해 주세요.");
        continue;
    }
    break;
}
if (n == 1 || n == 2)
    death.stop();
switch (n) {

case 1:
    penaltyOfDeath(p);
    // true = 마을로 돌아가는 선택
    this.result = true;
    break;

case 2:
    System.out.println("    프로그램을 종료합니다.");
    System.exit(0);

default:
    System.out.println("    20초동안 입력이 없어 종료합니다.");

    System.exit(0);
}

try {
    death.join();
} catch (InterruptedException e) {

    e.printStackTrace();
}
```

실행메서드

개선점

01

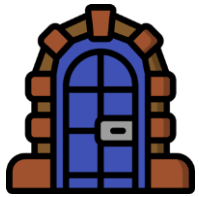
게임방법

02

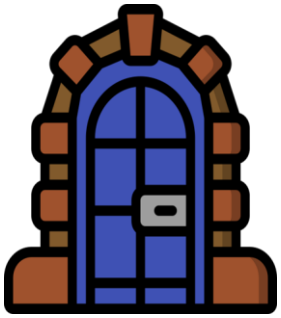
try/ catch

04

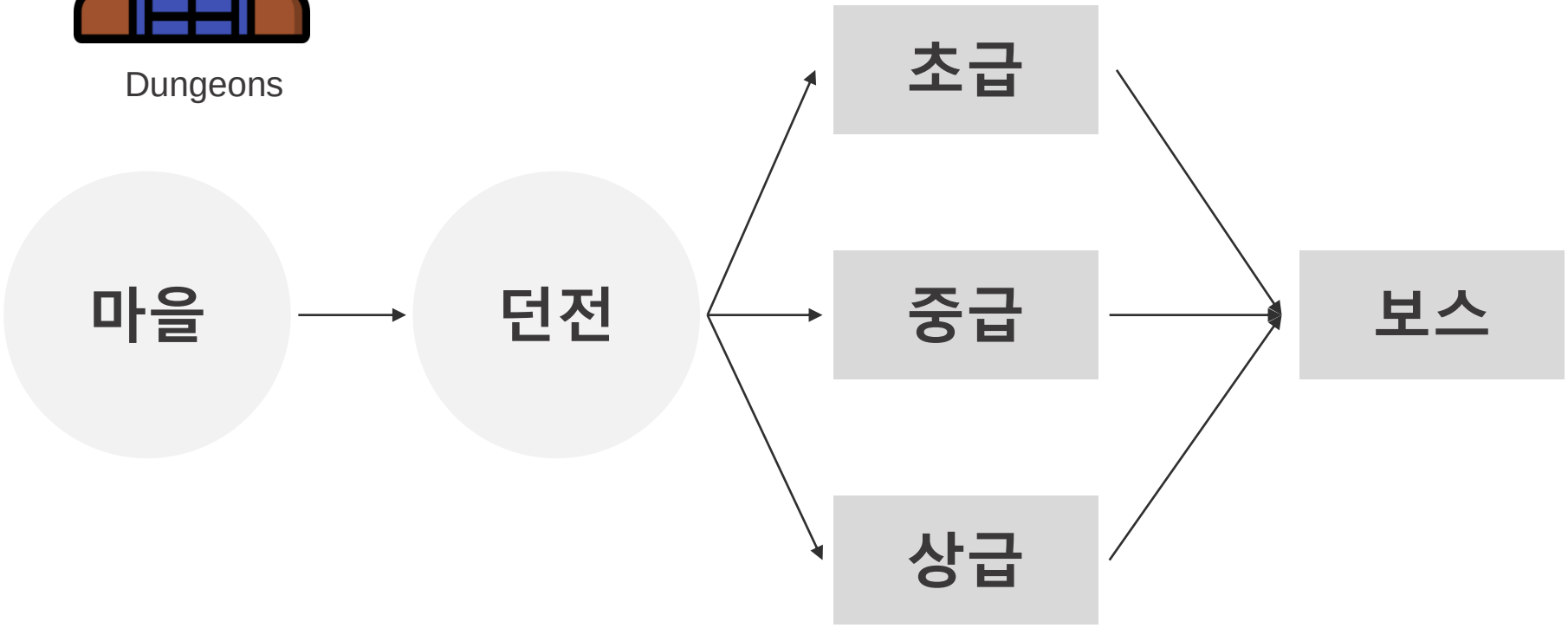
Dungeon

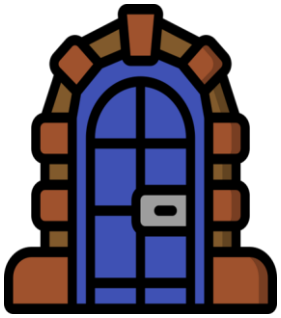


지용욱



Dungeons





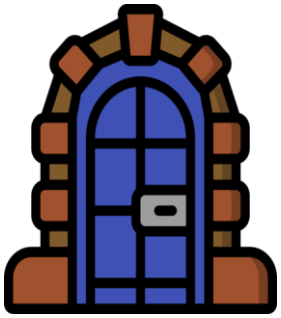
Dungeons

입장

```java
public interface DungeonIf {
    int EASY = 1;
    int NOMAL = 2;
    int HARD = 3;
    int BACK = 4;
}
```

```java
// 스테이지 선택
int stageChoice() {
    while(true) {
    System.out.println("    ================================");
    System.out.println("    |     던전을 선택해주세요       |");
    System.out.println("    |                              |");
    System.out.println("    |   1. 초급 던전 (권장 레벨 1 ~ 9)  |");
    System.out.println("    |   2. 중급 던전 (권장 레벨 8 ~ 15) |");
    System.out.println("    |   3. 상급 던전 (권장 레벨 14 ~ 30) |");
    System.out.println("    ================================");
    int num = 0;
    try {
    num = sc.nextInt();
    if(!(num>0&&num<4)) {
        BadNumberException e = new BadNumberException();
        throw e;
    }
    }catch(BadNumberException e) {
        System.out.println("숫자 입력이 잘못 되었습니다.");
        continue;
    }finally {
        sc.nextLine();
    }
    if (num == 2 && stage2Check()) {
        num = 4;
        return num;
    } else if (num == 3 && stage3Check()) {
        num = 4;
        return num;
    }
    return num;
    }
}
```

```java
// 스테이지 선택
boolean stage(Player p) {

    int num = stageChoice();

    switch (num) {
    case DungeonIf.EASY:
        result = stageEasy(p);
        p.skillInven.resetSkillChance();

        break;
    case DungeonIf.NOMAL:
        result = stageNomal(p);
        p.skillInven.resetSkillChance();

        break;
    case DungeonIf.HARD:
        result = stageHard(p);
        p.skillInven.resetSkillChance();

        break;
    case DungeonIf.BACK:
        result = true;
        break;
    }

    return result;
}
```

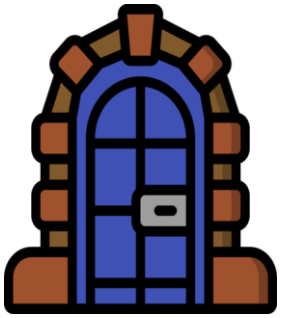


Dungeons

입장

```
boolean stage2Check() {

    boolean result = false;

    if (p.getStage2Count() == 0) {
        System.out.println("    +======================================+");
        System.out.println("    |        입장 권한이 없습니다.        |");
        System.out.println("    |    ※입장 조건 : 초급 던전의 보스 클리어    |");
        System.out.println("    +======================================+");
        result = true;
    }
    return result;
}

boolean stage3Check() {

    boolean result = false;

    if (p.getStage3Count() == 0) {
        System.out.println("    +======================================+");
        System.out.println("    |        입장 권한이 없습니다.        |");
        System.out.println("    |    ※입장 조건 : 중급 던전의 보스 클리어    |");
        System.out.println("    +======================================+");

        result = true;
    }
    return result;
}
```

```
// 스테이지 선택
int stageChoice() {
    while(true) {
    System.out.println("    ================================");
    System.out.println("    |     던전을 선택해주세요        |");
    System.out.println("    |                              |");
    System.out.println("    |   1. 초급 던전 (권장 레벨 1 ~ 9)  |");
    System.out.println("    |  2. 중급 던전 (권장 레벨 8 ~ 15)  |");
    System.out.println("    |  3. 상급 던전 (권장 레벨 14 ~ 30) |");
    System.out.println("    ================================");
    int num = 0;
    try {
    num = sc.nextInt();
    if(!(num>0&&num<4)) {
        BadNumberException e = new BadNumberException();
        throw e;
    }
    }catch(BadNumberException e) {
        System.out.println("숫자 입력이 잘못 되었습니다.");
        continue;
    }finally {
        sc.nextLine();
    }
    if (num == 2 && stage2Check()) {
        num = 4;
        return num;
    } else if (num == 3 && stage3Check()) {
        num = 4;
        return num;
    }
    return num;
    }
}
```

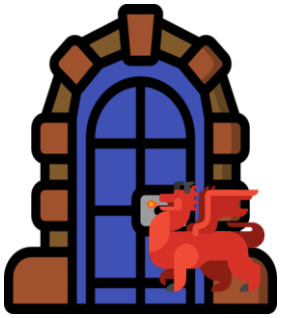


Dungeons

스테이지

```
boolean stageEasy(Player p) {

    System.out.println("    ----------------------------");
    System.out.println("        ★난이도 : 쉬움");
    System.out.println("    ----------------------------");

    for (int i = 1; i < 4; i++) {

        result = stage1(p, i);

        if (result) {
            break;
        }
    }
    return result;
}
```

```
boolean stage2(Player p, int num) {
    result = false;

    switch (num) {
    case 1:

        m = makeMonsters(num + 4);// 1 5 9
        break;
    case 2:

        m = makeMonsters(num + 4);
        break;
    case 3:

        m = makeMonsters(num + 4);
        break;
    }

    int win = b.choicePlayerMovement(m, p);

    if (num == 3 && win == 0) {
        e.rewordsOfVictory(p, m);

        System.out.println("    ----------------------------------------");
        System.out.println("    |   스테이지를 모두 클리어 하셨습니다.   |");
        System.out.println("    ----------------------------------------");
        bossStage(p, 2);
        result = true;
    } else if (win == 0) {
        e.rewordsOfVictory(p, m);
        System.out.println("    ---------------------------------");
        System.out.println("    |   다음 스테이지로 이동합니다.   |");
        System.out.println("    ---------------------------------");
    } else if (win == 1) {
        result = e.takeDie(p);

    } else if (win == 2) {
        System.out.println("    ========================= ");
        System.out.println("    |   마을로 돌아갑니다.|");
        System.out.println("    =========================");
        result = true;
    }
    return result;
}
```

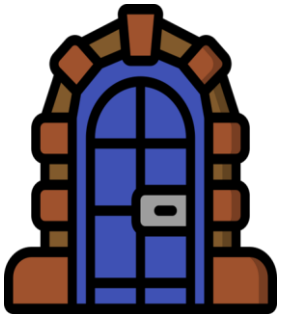


Dungeons

보스

```
boolean playBoss() {
    boolean result = false;
    System.out.println("    +=========================================================+");
    System.out.println("    |    던전의 보스는 상당히 강력하여 전투에서 도망 칠 수 없습니다.    |");
    System.out.println("    |    패배 시에는 일반적인 사망 패널티보다 더 많은 골드를 잃습니다.  |");
    System.out.println("    |    보스를 쓰러트렸을 경우에는 추가 경험치와 골드를 획득 할 수 있습니다.  |");
    System.out.println("    |        보스에 도전하시겠습니까? y or n      |");
    System.out.println("    +=========================================================+");
    String num = sc.nextLine();

    if (num.equals("y")) {

        System.out.println("    던전의 보스가 등장합니다. 행운을 빕니다.");
        result = true;
    }
    return result;
}
```

```
boolean bossStage(Player p, int num) {

    boolean win = false;

    if (playBoss()) {
        switch (num) {
        case 1:
            m = makeMonsters(num + 3);
            break;
        case 2:
            m = makeMonsters(num + 6);
            break;
        case 3:
            m = makeMonsters(num + 9);
            break;
        }

        int result = b.choicePlayerMovement(m, p);

        if (result == 0) {// result==1(몬스터 체력0이하일때=몬스터 죽었을 때)로 변경
            switch (num) {
            case 1:
                System.out.println("    ----------------------------------------");
                System.out.println("    |      초급 던전 보스를 처치 하셨습니다.  |");
                System.out.println("    |    상위 난이도 던전 입장 권한이 생겼습니다.   |");
                System.out.println("    |       추가 보상을 획득합니다.        |");
                System.out.println("    ----------------------------------------");
                p.setStage2Count(1);
                e.rewordsOfVictory(p, m);
                win = true;
                break;
            case 2:
                System.out.println("    ----------------------------------------");
                System.out.println("    |      중급 던전 보스를 처치 하셨습니다.  |");
                System.out.println("    |    상위 난이도 던전 입장 권한이 생겼습니다.   |");
                System.out.println("    |       추가 보상을 획득합니다.        |");
                System.out.println("    ----------------------------------------");
                p.setStage3Count(1);
                e.rewordsOfVictory(p, m);
                win = true;
                break;
            case 3:
                System.out.println("    ------------------------");
                System.out.println("    축하합니다!");
                System.out.println("    마지막 보스를 클리어 하셨습니다!");
                System.out.println("    ------------------------");
                e.rewordsOfVictory(p, m);
                win = true;
                break;
            }
            else if (result == 1) {
            win = e.takeDie(p);
        } else if (result == 2) {

            win = true;
        } else {
            System.out.println("    ------------------------");
```



Dungeons

개선점

- 중복 코드 메서드로 변경
- 메서드 타입 수정 필요
- 메서드명 수정 필요
- 예외처리

05

Events

김승연

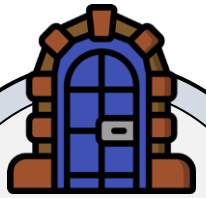
Battle

전투 결과 반환
win of lose

Rewards 계산

Penalty 계산

Events



Dungeon에게
계산값 전달

Events

```java
public class Events {

    Scanner input = new Scanner(System.in);
    boolean result = false;

    void rewordsOfVictory(Player p, Monster m) {

        // 0 = 패배 시, 1 = 승리 시
        // 승리 시
        System.out.println();
        System.out.println("    ┌++++++++++++++++++++++++++++++++++++++++++++++++++┐");
        System.out.println("              > 축하합니다." + m.getName() + "을(를) 물리쳤습니다!     ");

        if (p.getCurrentLevel() > (m.getWeakness() + 3)) {
            System.out.println("         [주의] 현재 레벨에 비해 몬스터가 약합니다!");
            System.out.println("         약한 몬스터에게선 획득할 수 있는 보상이 줄어듭니다.");
            p.setCurrentExp(p.getCurrentExp() + (m.getExpWorth() / 2));
            p.setGold(p.getGold() + (m.getGoldWorth() / 2));
            System.out.println();
            System.out.println(
                    "       > " + (m.getGoldWorth() / 2) + " 골드와 경험치" + (m.getExpWorth() / 2) + " 을(를) 획득했습니다.");
            System.out.println("    └++++++++++++++++++++++++++++++++++++++++++++++++++┘");
            p.checkLevelUp();

        } else if (p.getCurrentLevel() > (m.getWeakness() + 4)) {
            System.out.println("         [주의] 현재 레벨에 비해 몬스터가 너무 약합니다!");
            System.out.println("         약한 몬스터에게선 획득할 수 있는 보상이 줄어듭니다.");
            p.setCurrentExp(p.getCurrentExp() + (m.getExpWorth() / 3));
            p.setGold(p.getGold() + (m.getGoldWorth() / 3));
            System.out.println();
            System.out.println(
                    "       > " + (m.getGoldWorth() / 3) + " 골드와 경험치" + (m.getExpWorth() / 3) + " 을(를) 획득했습니다.");
            System.out.println("    └++++++++++++++++++++++++++++++++++++++++++++++++++┘");
            p.checkLevelUp();

        }

    }

}
```

```java
public Dungeon() {

    e = new Events();
    b = new Battle();
    p = new Player();
    m = new Monster();
    sc = new Scanner(System.in);
    runCount = false;
```

Dungeon 내에서 호출

Events

```
// 보스 사냥 성공시 추가 보상
void bonusRewordsKillBoss(Player p, Monster m) {

    p.setGold(p.getGold() + (m.getWeakness() * 200));
    p.setCurrentExp(p.getCurrentExp() + (m.getWeakness() * 50));
    System.out.println("\n\n");
    System.out.println("                                                  ");
    System.out.println("        보스 토벌 성공!          ");
    System.out.println("                                                  ");
    System.out.println("                                                  ");
    System.out.println("    >토벌 보상으로" + (m.getWeakness() * 200) + " Gold가 추가 지급됩니다.");
    System.out.println("    >토벌 보상으로" + (m.getWeakness() * 50) + " Exp가 추가 지급됩니다.");
    System.out.println("                                                  ");
    System.out.println("\n\n");
    p.checkLevelUp();
}
```

```
case 2:
    System.out.println("    ----------------------------------------");
    System.out.println("    |      중급 던전 보스를 처치 하셨습니다.  |");
    System.out.println("    |   상위 난이도 던전 입장 권한이 생겼습니다.   |");
    System.out.println("    |       추가 보상을 획득합니다.         |");
    System.out.println("    ----------------------------------------");
    p.setStage3Count(1);
    e.rewordsOfVictory(p, m);
    e.bonusRewordsKillBoss(p, m);
    win = true;
    break;
```

```
        System.out.println("
        System.out.println("
        System.out.println("
        System.out.println("
        System.out.println("
        System.out.println("
        System.out.println("
        System.out.println("\n\

        // 패널티를 지불할 충분한 골드가 없을 시

/
    }
```

Events

```
    // 패배 시
    void penaltyOfDeath(Player p) {

        System.out.println("    ┼═══════════════════════════════════════════┼");
        System.out.println("      쓰러져있던 플레이어를 마을로 긴급 이송합니다. : 이송 비용 발생");
        System.out.println();

        // 패널티를 지불할 충분한 골드가 있을 시
        if ((p.getGold() - (p.getCurrentLevel() * 100)) > 0) {
            p.setGold(p.getGold() - (p.getCurrentLevel() * 100));
            p.invenCurrentHealth = p.invenCurrentHealth + (int) (p.invenMaxHealth * 0.5);
            System.out.println("    ┌──────────────────────────────────────┐");
            System.out.println("    >이송 비용으로 " + (p.getCurrentLevel() * 100) + " Gold가 청구됩니다.   |");
            System.out.println("    >남은 골드 : " + p.getGold() + "            |");
            System.out.println("    >최대 체력의 절반이 회복되었습니다.  |");
            System.out.println("    └──────────────────────────────────────┘");

            // 패널티를 지불할 충분한 골드가 없을 시
        } else if ((p.getGold() - (p.getCurrentLevel() * 100)) < 0) {
            System.out.println("    이송 비용을 지불할 소지금이 부족합니다.");
            System.out.println("    가장 최근에 저장한 세이브 파일을 불러옵니다.");
            p.loadPlayer();

        }
/
    }
```

Events

```
            try {
                n = Integer.parseInt(choice);
                if (!(n > 0 && n < 3)) {
                    System.out.println("    정상적인 메뉴 선택이 아닙니다.\n메뉴를 다시 선택해주세요.");
                    continue;
                }

            } catch (NumberFormatException e) {
                System.out.println("    잘못입력하셨습니다. 다시 선택해 주세요.");
                continue;
            } catch (InputMismatchException e) {
                System.out.println("    잘못입력하셨습니다. 다시 선택해 주세요.");
                continue;
            }
            break;
        }
        if (n == 1 || n == 2)
            death.stop();
        switch (n) {

        case 1:
            penaltyOfDeath(p);
            // true = 마을로 돌아가는 선택
            this.result = true;
            break;

        case 2:
            System.out.println("    프로그램을 종료합니다.");
            System.exit(0);

        default:
            System.out.println("    20초동안 입력이 없어 종료합니다.");

            System.exit(0);
        }
```

```
            try {
                death.join();
            } catch (InterruptedException e) {

                e.printStackTrace();
            }
        }
        return result;
    }

}
```

현재 20초 경과 후
자동 종료가 되지 않는 문제 개선필요

J. J. J 's TEXT RPG

파일(F) 편집(E) 서식(O) 보기(V) 도움말

Town

Main 메서드와 Dungeon을 연결해주는
중간 통로 역할

```
public class Town {

    Scanner input = new Scanner(System.in);
    Dungeon d = new Dungeon();
    Store store = new Store();
    In tory inven = new Inventory();

        town() throws InterruptedException {
```

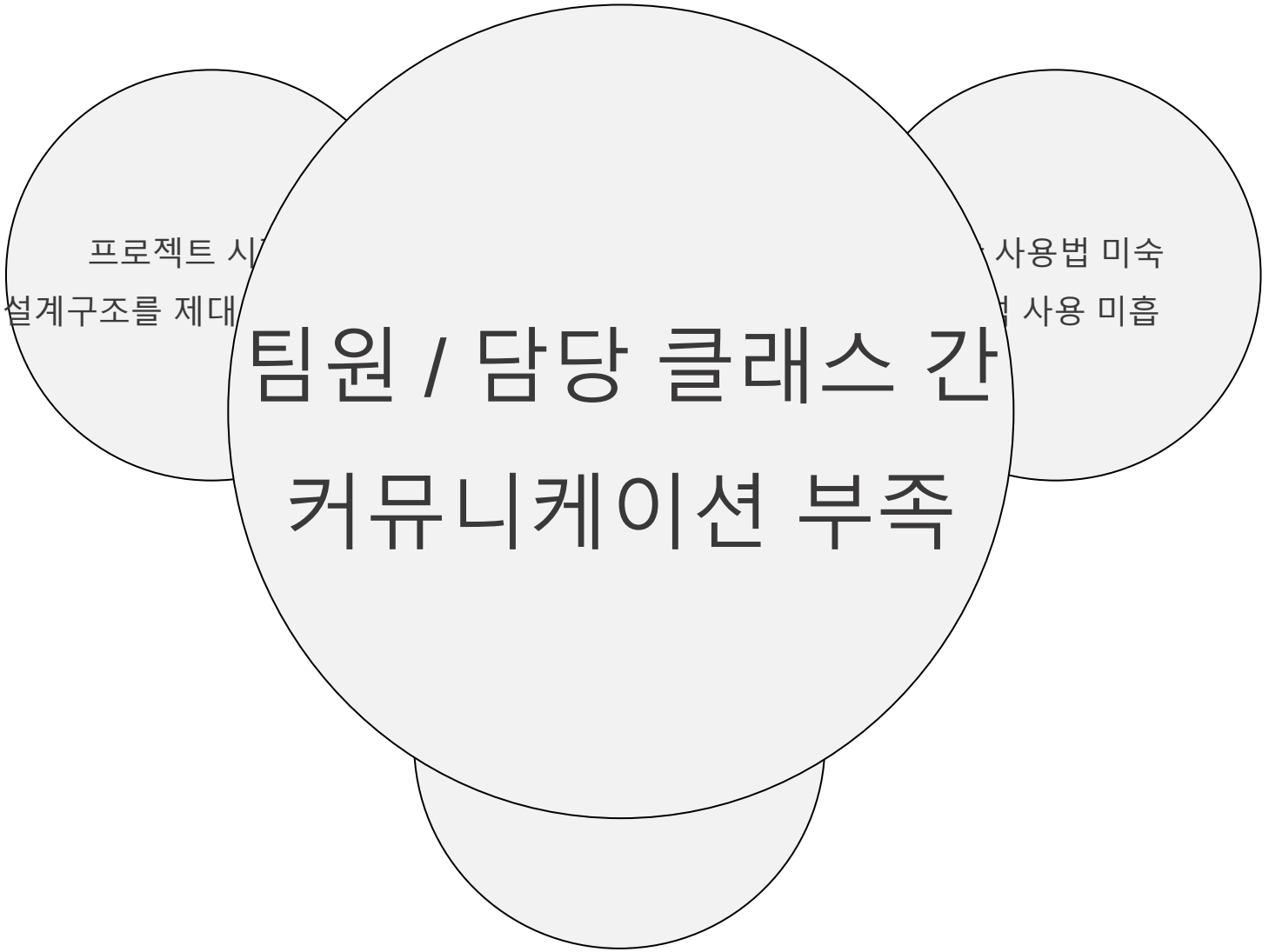
```
++++    JAVA TOWN   +++++++++");
                    |");
    무엇을 하시겠습니까?        |");
                    |");
  1. 던전 가기          |");
  2. 캐릭터 정보           |");
  3. 장비창            |");
  4. 상점 이용          |");
  5. 여관 이용 (체력 회복 가능)    |");
  6. 현재 데이터 저장하기      |");
  7. 이전 데이터 불러오기      |");
  8. 게임 종료          |");
                    |");
    ++++++++++++++++++++++++++++++++++++++++");
("\n\n\n\n\n\n\n");
input.nextInt();
```

```
public static void main(Str        throws InterruptedException {
    @SuppressWarnings("res
    Scanner sc = new Scar      (System.in);
    int choice;
    Town t = new Town();
    System.out.println("\n");
    System.out.println("          Adventure In The JAVA");
    System.out.println();
```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Events

```
System.out.println("Events 클래스 작업 시 힘들었던 점 : "
        + "플레이어와 몬스터, 던전의 데이터를 모두 사용하고 계산 후 다시 반환해줘야하는 클래스 특성상"
        + "어떤 클래스에서 호출/사용해야 정보를 받아 처리한 결과값을 잃지 않고 반환해 줄 수 있는지에 대해"
        + "구조 설계를 정확히 하지 않고 작업을 시작 -> 변수와 메서드들의 연결과 반환값이 저장이 안되는 문제 발생"
        + "사용되는 모든 변수와 메서드를 다시 확인하고  일일히 연결하는 과정이 매우 힘들었음.");


System.out.println("Events 클래스 작업 후 개선사항 및 느낀점 : "
        + "퀘스트 기능의 추가나 던전 내에서의 돌발상황 등의 다양한 재미를 줄 수 있는 이벤트 기능 추가"
        + "팀원간 소통이 충분하다고 생각했는데 그것마저도 아주 부족했다고 느끼게 된 팀프로젝트");
```


프로젝트 시
설계구조를 제대
사용법 미숙
사용 미흡
팀원 / 담당 클래스 간
커뮤니케이션 부족

System.out.println
(“ Thank you! :) ”);